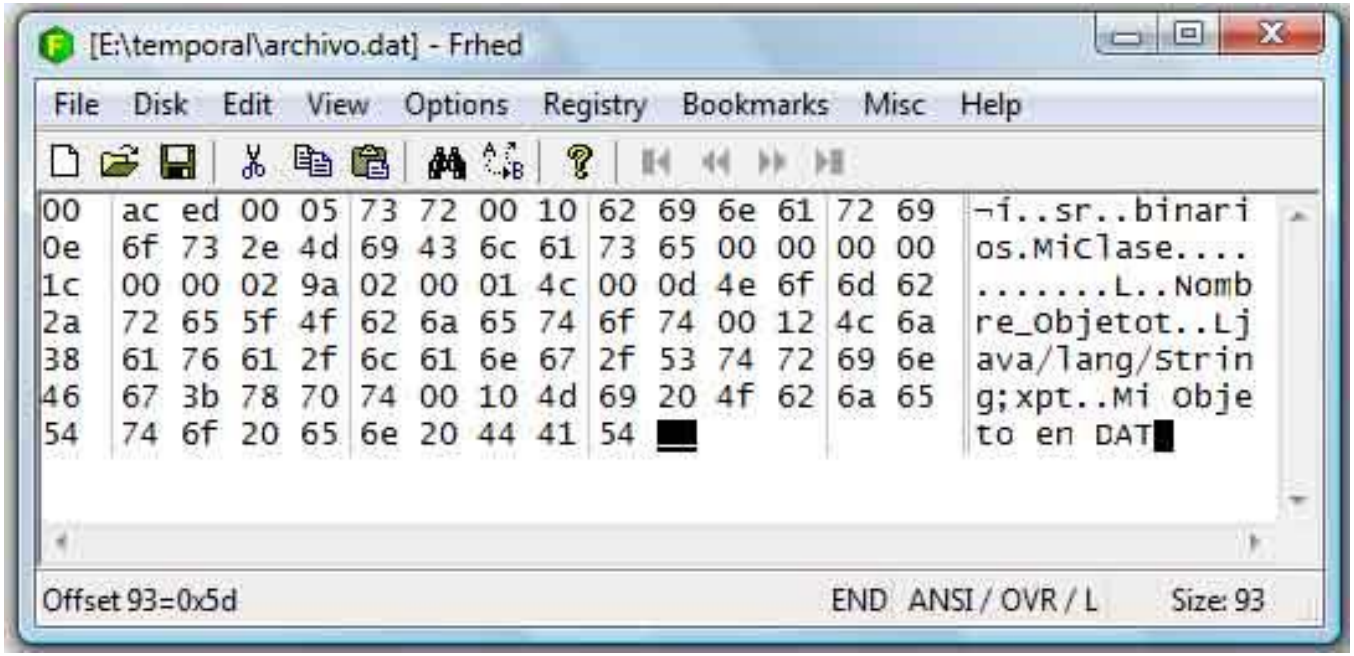

ARCHIVOS BINARIOS EN C++

UTEC
UNIVERSIDAD DE INGENIERÍA Y TECNOLOGÍA

Definición de un archivo binario

- Contienen información de cualquier tipo de dato pero en formato binario.
- Los datos se almacenan tal y como se almacenan en la memoria principal, no se convierten en caracteres para almacenarlos
- Por lo tanto, dicha información almacenada no es legible al ojo humano por lo que se requiere de una programa para poder procesarlo.

UTEC
UNIVERSIDAD DE INGENIERÍA Y TECNOLOGÍA

Definición de un archivo binario

- Puede guardarse datos primitivos: int, float, etc
- Pero generalmente cada elemento del que se quiere guardar información se almacena en un registro (*struct o class*).
- Si los elementos se almacenan con registros, se puede acceder directamente al *n-ésimo* elemento sin leer los $n - 1$ anteriores.
- Los ficheros de texto son ficheros de acceso secuencial, y los binarios son ficheros de acceso directo (o acceso aleatorio)

UTEC
UNIVERSIDAD DE INGENIERÍA Y TECNOLOGÍA

Declaración de variables, apertura y cierre de ficheros binarios

- Declaración de variables: como en los ficheros de texto:

```
ifstream fbl;    // fichero para lectura
ofstream fbe;    // fichero para escritura
```

- Apertura del fichero: hay que añadir "*ios::binary*"

```
fbl.open("mifichero.dat", ios::in | ios::binary);
fbe.open("mifichero.dat", ios::out | ios::binary);
```

Declaración de variables, apertura y cierre de ficheros binarios

- Cierre del fichero: como en los ficheros de texto, con *close*:

```
fbl.close();
```

- Otros modos de apertura:

lectura/escritura	`ios::in | ios::out | ios::binary`
añadir al final	`ios::out | ios::app | ios::binary`

UTEC
UNIVERSIDAD DE INGENIERÍA Y TECNOLOGÍA

Lectura de un fichero binario

```
typedef struct { ... } TIPOCIUDAD;
...

  TIPOCIUDAD ciudad;

  fbl.open("mifichero.dat",ios::in | ios::binary);
  if (fbl.is_open())
  {
     fbl.read((char *)&ciudad, sizeof(ciudad));
     while (!fbl.eof())
     {
        // procesar 'ciudad'

       fbl.read((char *)&ciudad, sizeof(ciudad));
     }
     fbl.close();
  }
```

UTEC
UNIVERSIDAD DE INGENIERÍA Y TECNOLOGÍA

Acceso directo a un elemento

- La posición de un elemento en el fichero se puede calcular en función del tamaño (*sizeof*) de lo que hay antes.

```
if (fbl.is_open())
{
  // posicionar para leer el tercer elemento
  fbl.seekg ( (3-1)*sizeof(ciudad), ios::beg);
  fbl.read( (char *)&ciudad, sizeof(ciudad) );
  ...
}
```

- Otras referencias para la posición:
  - *fbl.seekg(pos, ios::cur)* desde la posición actual
  - *fbl.seekg(pos, ios::end)* desde el final del fichero

OJO: pos puede ser negativo

UTEC
UNIVERSIDAD DE INGENIERÍA Y TECNOLOGÍA

Escritura de un fichero binario

```
typedef struct { ... } TIPOCIUDAD;
...

  TIPOCIUDAD ciudad;
  ofstream fbe;

  fbe.open("mifichero.dat",ios::out | ios::binary);
  if (fbe.is_open())
  {
     // rellenar 'ciudad'

     fbe.write((const char *)&ciudad, sizeof(ciudad));
     ...
  }
```

UTEC
UNIVERSIDAD DE INGENIERÍA Y TECNOLOGÍA

Acceso directo a un elemento para escribir

Hay que utilizar el método *seekp* (posicionamiento para escritura) en lugar de la *seekg* (para lectura).

```
if (fbe.is_open())
{
  // posicionar para escribir el tercer elemento
  fbe.seekp ( (3-1)*sizeof(ciudad), ios::beg);
  fbe.write( (const char *)&ciudad, sizeof(ciudad) );
  ...
}
```

ATENCIÓN: si la posición a la que se va a ir con *seekp* no existe en el fichero, se alarga el fichero para que se pueda escribir.

UTEC
UNIVERSIDAD DE INGENIERÍA Y TECNOLOGÍA

Escritura/lectura de registros con cadenas de caracteres

Si se desea almacenar un registro en un fichero binario que contenga una cadena de caracteres, se debe utilizar un vector de caracteres, nunca un *string*. Puede ser necesario recortar el *string*:

```
char cad[tcREG];
string s;

...
strncpy(cad,s.c_str(),tcREG-1);
cad[tcREG-1]='\0'; // strncpy no pone el \0 si no aparece
```

UTEC
UNIVERSIDAD DE INGENIERÍA Y TECNOLOGÍA

Información sobre la posición actual

La posición actual del puntero de lectura (en bytes) se puede ver mediante la función *tellg*, y la de escritura mediante *tellp*. Ejemplo:

```
 // Colocamos el puntero de lectura al final
f.seekg(0,ios::end);

// Calculamos el numero de registros del fichero
cout << f.tellg()/sizeof(miRegistro) << endl;
```

UTEC
UNIVERSIDAD DE INGENIERÍA Y TECNOLOGÍA

Errores de lectura/escritura

- Se producen tanto en ficheros de texto como en ficheros binarios.
- Es raro que se produzcan, pero hay que tenerlos previstos.
- Es recomendable comprobar que no hay errores después de cada lectura/escritura.
- Se debe utilizar el método fail (aunque hay otras formas):

```
if (fi.fail() && !fi.eof()) // error de lectura
      ...
```

UTEC
UNIVERSIDAD DE INGENIERÍA Y TECNOLOGÍA

Errores de lectura/escritura

```
if (fi.is_open())
{
   bool error=false;
   getline(fi,s);
   if (fi.fail() && !fi.eof()) error=true;
   while (!error && !fi.eof())
   {
      // hacer algo con 's'

      getline(fi,s);
      if (fi.fail() && !fi.eof()) error=true;
   }

   if (error)
      // mensaje de error
   fi.close();
}
```